Algorithms for Programming Contests
SS17 - Week 7

Chair for Foundations of Software Reliability and Theoretical Computer Science, TU München

Stefan Jaax, Philipp Meyer, Christian Müller

Welcome to our practical course! This problem set is due by

Thursday, 29.06.2017, 6:00 a.m.

Try to solve all the problems and submit them at

`https://judge.in.tum.de/conpra/`

This week's problems are:

A **Exam Preparation** 1

B **Street Lights** 3

C **Woodchucking** 5

D **Tour d'Absurdistan** 7

E **Mice** 9

The following amount of points will be awarded for solving the problems.

Problem	A	B	C	D	E
Difficulty	easy	easy	medium	medium	hard
Points	4	4	6	6	8

If the judge does not accept your solution but you are sure you solved it correctly, use the "request clarification" option. In your request, include:

- the name of the problem (by selecting it in the subject field)
- a verbose description of your approach to solve the problem
- the time you submitted the solution we should judge

We will check your submission and award you half the points if there is only a minor flaw in your code.

If you have any questions please ask by using the judge's clarification form.

This page is intentionally left blank.

Problem A
Exam Preparation

Lea heard that at TUM (Thomas Underwood University Markistan - Thomas Underwood was a famous scientist, but he never got well-known beyond the borders of Markistan) students are not allowed to take any material to their exams except a hand-written single-sided sheet of paper. Therefore, all students try to write as much as possible on their cheat sheets. There is obviously a limit in font size which means they have to decide which information helps the most. To make things more complicated, each professor allows different sizes of the cheat sheets: Some of them are very generous, whereas others only allow for stamp-sized sheets.

Lea is unsure whether she wants to apply at TUM, but if she does, she wants to master the cheat sheet system. For each lecture she knows all topics covered and the number of characters that fit on a cheat sheet of the allowed size. For each topic, she also knows the number of characters one piece of information needs and how useful they are (Lea has an integer score system where a higher score means more useful). For each topic Lea will find an unlimited supply of information units and each of them will have the same length and score. It would be sufficient to add many pieces of information for only a few topics, but Lea wants the overall score of her cheat sheet to be as big as possible.

After all, Lea knows that understanding the lectures is much more useful than a perfect cheat sheet. That is why she does not want to spend too much time creating the cheat sheets. Lea is happy if she has information with at least half of the best possible score on her cheat sheet. Help her to choose what to add to it.

Input

The first line of the input contains an integer t, the number of lectures. t lectures follow, each of them separated by a blank line.

Each lecture starts with a line containing two integers: m, the number of characters that fit on the allowed cheat sheet, and n, the number of topics covered. n lines describing the topics follow. The i-th line contains two integers l_i and s_i where l_i is the length of a piece of information for this topic and s_i is its score.

Output

For each test case, output one line containing "Case #i: x" where i is its number, starting at 1, and x is a space-separated list of topics to be added (topics may appear several times in this list). The sum of their lengths should be at most m and the sum of their scores should be at least half of the optimal value.

Constraints

- $1 \leq t \leq 20$
- $1 \leq n \leq 100$
- $1 \leq m \leq 10000$
- $1 \leq l_i \leq 10000$ for all $1 \leq i \leq n$
- $1 \leq s_i \leq 10000$ for all $1 \leq i \leq n$

Sample Input 1	Sample Output 1

<pre>6
10 2
3 5
1 1
 	
10 3
3 7
2 8
7 5
 	
17 4
2 4
3 3
5 2
4 7
 	
8 2
2 4
3 9
 	
25 1
2 1
 	
15 5
2 1
4 5
6 7
8 9
10 11</pre>
	<pre>Case #1: 1 1 1 2
	Case #2: 2 2 2 2 2
	Case #3: 1 1 1 1 1 1 1 1
	Case #4: 2 2 1
	Case #5: 1 1 1 1 1 1 1 1 1 1 1 1
	Case #6: 2 2 2 1</pre>

Sample Input 2	Sample Output 2

<pre>5
80 3
2 1
40 20
20 11
 	
10 3
100 10000
50 100
20 300
 	
16 4
8 8
3 1
7 5
8 7
 	
50 3
17 70
15 35
13 33
 	
13 4
2 1
8 3
5 4
7 6</pre>
	<pre>Case #1: 3 3 3 3
	Case #2:
	Case #3: 1 1
	Case #4: 1 1 3
	Case #5: 4 3</pre>

Problem B
Street Lights

During another of Lea's visits to her uncle in Chaosville, she realizes that the streets of Chaosville are in horrible condition. Driving at night is really dangerous: you never know when you will hit a pothole, because most street lights are not working and you see close to nothing. One day, Lea decides to visit the mayor's office to talk about the bad lighting on the streets. The mayor explains the city's problem to Lea: indeed, there are many street lights, but erected in an irregular fashion at the side of every street. And most of them are switched off due to severe budget cuts. Not willing to give up, Lea offers to come up with a plan that specifies how many of the street lights should be switched on to illuminate the main street, at least. The mayor agrees, but only if she can provide him with a solution that needs to switch on only as few street lights as possible.

Input

The first line of the input contains an integer t. t test cases follow, each of them separated by a blank line.

Each test case starts with three integers ℓ n d, the length of the main street ℓ, the number of street lights on main street n, and the radius of the light cone of each street light d, which indicates how far each light shines. Then, another line follows, consisting of n integers p_1 p_2 $\dots$ p_n, describing the positions of the street lights.

Output

For each test case, output one line containing "Case #i: x" where i is its number, starting at 1, and x is either the smallest number of street lights that are needed to illuminate the whole main street (which goes in a straight line from 0 to ℓ), or "impossible" if there is no way to illuminate the whole street.

To illuminate the whole main street, there should be light from at least one street light at every point on the main street between 0 and ℓ. The boundary of each light cone is considered to be illuminated as well. In particular, this means that a point on the street is illuminated if two light cones touch there, they do not need to intersect. See the sample data explanation.

Constraints

- $1 \leq t \leq 20$
- $1 \leq \ell \leq 50000$
- $0 \leq n \leq \min(1000, \ell + 1)$
- $0 \leq d \leq 1000$
- $0 \leq p_i \leq \ell$ for all $1 \leq i \leq n$
- $p_i \neq p_j$ for all $1 \leq i, j \leq n$

Sample Data Explanation

In the first sample case, three street lights need to be switched on. Note that these may also be three different street lights, for example switching on the first, third, and sixth street light works as well. The second and third sample cases are impossible as there is always at least one section of the street that cannot be illuminated. In the fourth sample case, the street can be illuminated on its entire length by switching on all street lights. The points at which the light cones meet between the street lights are illuminated as well.

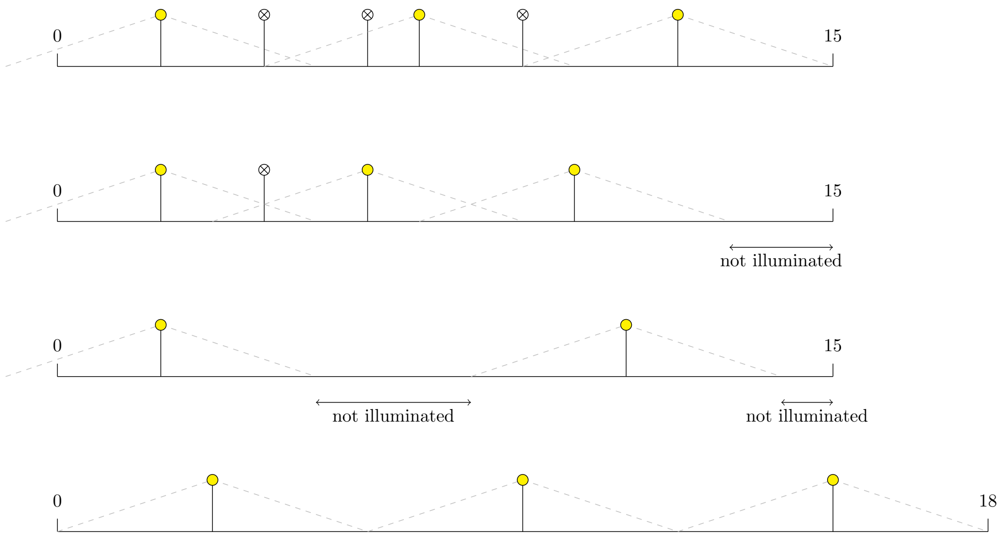

Figure B.1: Visualization of the first four sample cases.

Sample Input 1

```
2
15 6 3
2 4 9 7 6 12

15 4 3
10 4 6 2
```

Sample Output 1

```
Case #1: 3
Case #2: impossible
```

Sample Input 2

```
7
15 2 4
2 11

18 3 3
3 15 9

21 14 4
14 8 10 7 20 21 3 6 18 15 16 12 9 5

9 9 3
1 2 7 3 8 0 9 6 4

14 12 5
2 3 12 10 8 1 7 0 5 13 14 11

23 14 3
2 9 7 11 5 19 8 0 23 14 15 6 21 10

14 8 4
14 8 0 6 13 9 10 1
```

Sample Output 2

```
Case #1: impossible
Case #2: 3
Case #3: 3
Case #4: 2
Case #5: 2
Case #6: 5
Case #7: 3
```

Problem C
Woodchucking

You may remember Lea's friend Nick, the biologist. Recently, he decided to do a study at a saw mill, *Jack Lumber Inc.*, a business producing lumber. He decided to do his study on the possible productivity increase of local industries by using rodents (woodchucks, mostly). So for a few weeks, Nick got in touch with his inner lumberjack and hacked away at everything that even remotely resembled a tree (the lumberjacks quickly learned not to come too close to Nick when he was armed with an axe). Once a month, the lumberjacks had to heave all the trees onto large conveyor belts at the saw mill, where they were cut up into small pieces by huge disk saws, ready to be shipped to whoever wanted to build barstools, wooden swords, toothpicks or whatever else. There were a lot of disk saws and conveyor belts and every tree took sawing time according to its height. This meant nobody knew on which conveyor the next tree had to be put to minimize the total sawing time. So these days were mostly ending in chaos and the buzzing and sawing took quite a while.

As a scientist, Nick could not resist to try to help them and quickly devised a computer program that computed the optimal distribution of trees to the saws. It took ages to compute the value, but now the lumberjacks could chuck down the trees, then sit around for a day and then Nick guided them to cut the trees up much faster.

Unfortunately, Handsome Jack, the boss at *Jack Lumber Inc.*, really did not like the fact that his employees sat around, not chucking wood, waiting for Nick's program to spurt out some numbers. Nick tried to reason with him, but it was no use. They had to go back to the old system until Nick came up with a better idea.

Can you help him devise a faster computation that cuts up the trees in at most 50% more time than his previous solution?

Input

The first line of the input contains an integer t. t test cases follow, each of them separated by a blank line. Each test case consists of a line containting two integers n and m. n is the amount of trees the lumberjacks felled during the course of the month, m is the amount of disk saws available. n lines follow, each containing a single integer l_i, the time it takes to cut up tree i.

Output

For each test case, output one line containing "Case #i: s" where i is its number, starting at 1, and s is at most 50% more than the minimal amount of time it takes to cut up every tree. Each line of the output should end with a line break.

Constraints

- $1 \leq t \leq 20$
- $1 \leq n \leq 2 \cdot 10^5$
- $1 \leq m \leq 1000$
- $1 \leq l_i \leq 10^5$

Sample Input 1

```
2
3 2
2
8
4

4 3
1
1
2
4
```

Sample Output 1

```
Case #1: 8
Case #2: 4
```

Sample Input 2

```
5
4 5
1
3
7
4

4 2
4
9
5
5

3 2
10
9
9

4 4
6
4
8
7

3 4
8
4
10
```

Sample Output 2

```
Case #1: 7
Case #2: 13
Case #3: 18
Case #4: 8
Case #5: 10
```

Problem D
Tour d'Absurdistan

Once a year, the people of Absurdistan invite sportspersons from all around the world to compete in a long muscle-powered segway race that spans over the course of three weeks. The race always goes through most of the larger cities in Absurdistan – for example, Preposteropolis, Ridiculousia, and Ludicrous City come to mind – but the routes it will take in between these cities and the order in which the cities are visited are not fixed yet. Because Lea's reputation for solving these kinds of problems has reached even to Absurdistan, she is asked to do it. But there are three important rules which the tour must fulfill to be officially called "*Tour d'Absurdistan*": Firstly, each city must be visited, but only once; secondly, the total distance should be as low as possible, and thirdly, the tour must end in the same city in which it started. Can you tell Lea how to do it?

Input

The first line of the input contains an integer t. t test cases follow, each of them separated by a blank line.

Each test case starts with an integer n, the number of cities (indexed from 1 to n). n lines follow describing the distances between the cities, each containing n integers $l_{i,1}, \ldots, l_{i,n}$ with $l_{i,j}$ being the distance from city i to city j. Note that connections are undirected, i.e., $l_{i,j} = l_{j,i}$ and $l_{i,i} = 0$.

Output

For each test case, output one line containing "Case #i: x" where i is its number, starting at 1, and x the optimal tour given as a space-separated sequence of all cities. As this is a tour where every city appears only once, the order in which the cities are given is not important as the sequence can be shifted such that the first city appears as the first element in the sequence. It is implicitly assumed that the tour goes around, i.e., after leaving the last city, the competitors race back to the finish line in the first city. Each line of the output should end with a line break.

Constraints

- $1 \leq t \leq 20$
- $1 \leq n \leq 8$
- $1 \leq l_{i,j} \leq 1000$ for all $1 \leq i, j \leq m, i \neq j$
- The distance from city i to city i will always be 0.

Sample Input 1

```
2
4
0 11 8 8
11 0 11 3
8 11 0 10
8 3 10 0

4
0 10 12 5
10 0 8 13
12 8 0 7
5 13 7 0
```

Sample Output 1

```
Case #1: 1 3 2 4
Case #2: 1 2 3 4
```

Sample Input 2	Sample Output 2
10 4 0 7 4 2 7 0 1 4 4 1 0 9 2 4 9 0 4 0 7 3 3 7 0 3 2 3 3 0 7 3 2 7 0 3 0 2 2 2 0 2 2 2 0 3 0 5 9 5 0 6 9 6 0 4 0 9 1 7 9 0 2 10 1 2 0 9 7 10 9 0 3 0 8 8 8 0 2 8 2 0 3 0 2 1 2 0 4 1 4 0 5 0 8 8 3 4 8 0 7 4 1 8 7 0 10 1 3 4 10 0 7 4 1 1 7 0 4 0 10 6 3 10 0 6 2 6 6 0 8 3 2 8 0 3 0 2 1 2 0 1 1 1 0	Case #1: 1 3 2 4 Case #2: 1 3 2 4 Case #3: 1 2 3 Case #4: 1 2 3 Case #5: 1 3 2 4 Case #6: 1 2 3 Case #7: 1 2 3 Case #8: 1 3 5 2 4 Case #9: 1 3 2 4 Case #10: 1 2 3

Problem E
Mice

Recently Lea read a book about, among other things, the creation of the earth. In this scientifically accurate work it was detailed that the earth was a construction ordered by mice. This left Lea baffled: could it really be that humans are inferior to mice? Only one way to find out!

Lea wants to make sure that humans are superior by comparing their DNA to the DNA of an obviously superior being, dolphins. Should she find out that they are closely related, it is clear that mice are no match for humans.

Comparing DNA is done by scoring: Lea has some DNA sequences from humans and some from dolphins. A sequence is a string consisting only of the letters A,C,T and G. Two equal length sequences are scored with a 4×4 scoring matrix whose rows and colums are labelled with A,C,T and G. An entry in row A and column T is then called $score(A, T)$. The sequences are scored by computing the score of the first letters of each sequence, the second letters, and so on, and summing up the result. For example, the sequences ATA and CGT would be scored as $score(A, C) + score(T, G) + score(A, T)$.

The matrix must have the following properties:

- All entries must be integers between -10 and 10, inclusively.
- It must be symmetric ($score(x, y) = score(y, x)$ for all $x, y \in \{A, C, T, G\}$).
- Diagonal entries must be positive ($score(x, x) > 0$ for all $x \in \{A, C, T, G\}$).
- The sum of all 16 entries must be 0.

Of course Lea wants to choose the matrix such that the sum of the scores she gets when comparing every human DNA sequence to every dolphin DNA sequence is maximal. Can you compute the maximal score that she can achieve?

Input

The first line of the input contains an integer t. t test cases follow, each of them separated by a blank line.

The first line of each test case contains two integers n m, the number of human and dolphin DNA sequences. The next n lines contain one human DNA sequence each. The last m lines contain a dolphin DNA sequence each.

Output

For each test case, print a line containing "Case #i: x" where i is its number, starting at 1, and x is the maximal score that can be reached. Each line of the output should end with a line break.

Constraints

- $1 \leq t \leq 20$
- $1 \leq m, n \leq 200$
- A sequence consists of at most 100 letters of A,C,T and G.
- For each test case, all sequences have equal length.

Sample Input 1	Sample Output 1
7 1 1 ACAC ATGC 2 2 AAA CTG TCC GGT 3 3 ACACACTGGTCTATACTTCG ACTCGCTGAAGTTAATTACC ACTGTGCGTCCAAGGGTAAT ATCGGACAGATCACGCCCCT ATGGTGATATTCAATGCTGT CATATCGTACTAGGGTAAAG 2 5 CTATTTAGTT AAACTGTAAT GATTGATTTG CTAGAGACCA ATTGCTCCCG GAGTACTAAA TGCCGTTCTT 3 2 GGACGCG ATTCGAT ATAGCGA TCACCCG CCCACAG 4 5 TACG GGAG ACCC GTCA GCAC GCGG GAAG GGTA ATCT 3 2 CGTATGTCCCGCGA GCAAGTGCGCTTTG CAGGTTGGGGTCAA GGTAAGGGGAAAAC ATAAGCTTGCTCTC	Case #1: 40 Case #2: -4 Case #3: 322 Case #4: 363 Case #5: 147 Case #6: 236 Case #7: 149